What is Big Data?

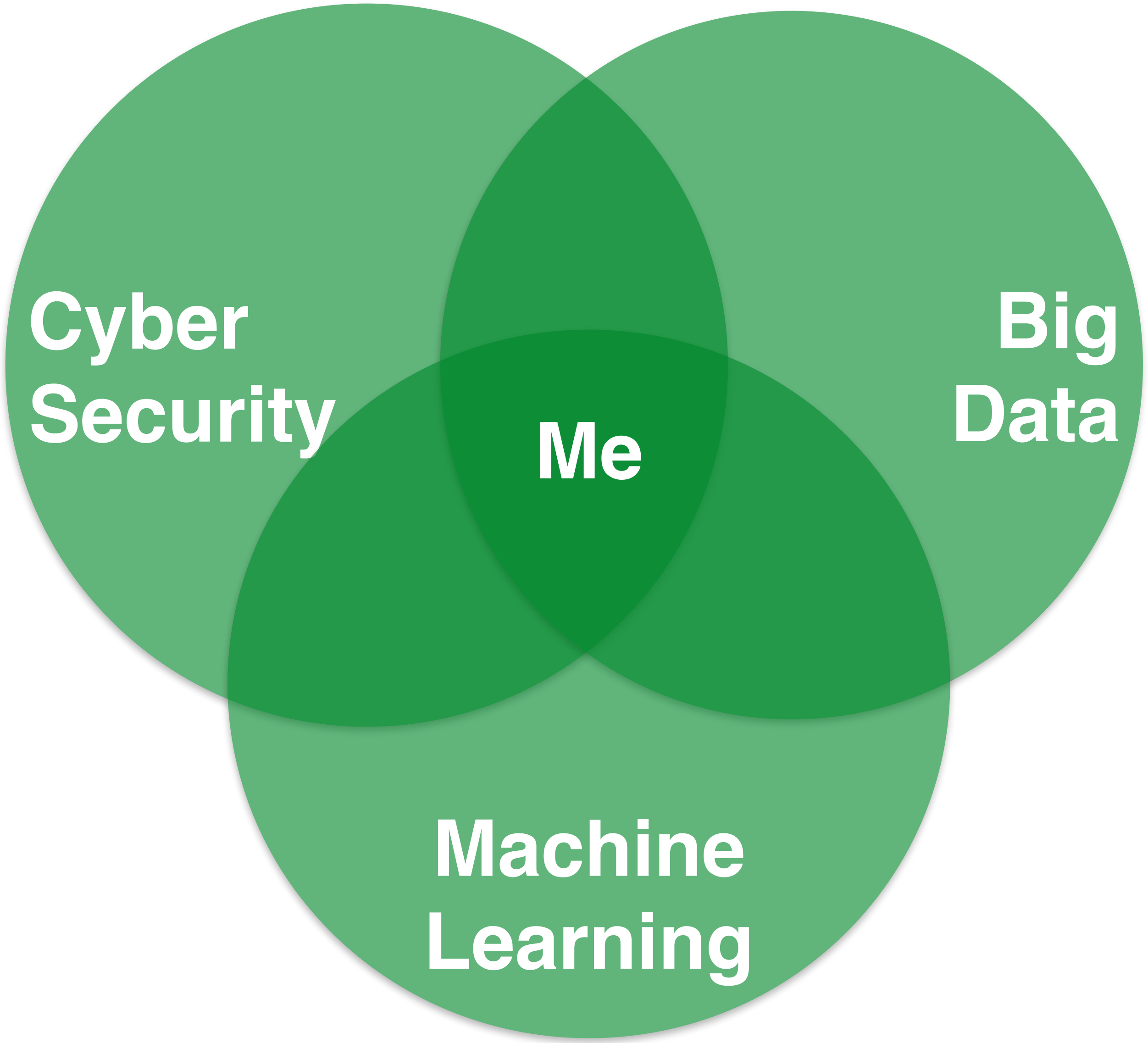
Concepts, Ideas and Principles

Hitesh Dharamdasani

INFORMANT
NETWORKS

# whoami

- Security Researcher, Malware Reversing Engineer, Developer
- GIT > George Mason > UC Berkeley > FireEye > On Stage
- Building Data-driven Network Security Products at Informant Networks
- Hiring!



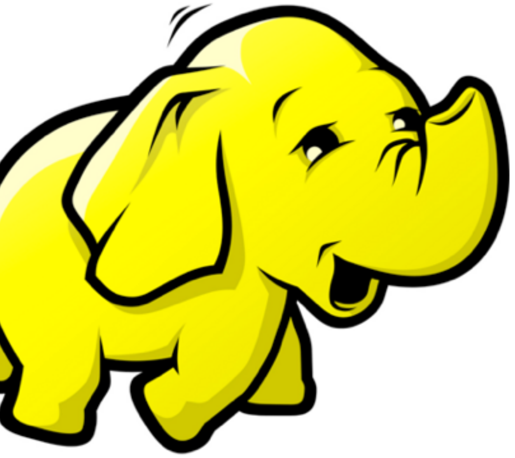
Agenda

- What is Big Data?
- What do we mean by a Distributed system?
- How does HDFS work?
- What is Map Reduce
- When is MapReduce supposed to be used
- Spark

SIGINT for questions

HIVE

mahout



Scalding

Spark

APACHE
HBASE

What is BigData?

- Very broad term

- Usually means that computation code has become minuscule compared to error handling and synchronisation

- Data is heterogeneous

- 100 GB is not yet BigData. 10+ TB. Probably?

- MapReduce won sorting challenge. 102 TB in 72 mins with 2100 nodes

Why so much data?

- One tweet is an anecdote
  - But a million tweets may signal important trends
- One person’s product review is an opinion
  - But a million reviews might uncover a design flaw
- One person’s diagnosis is an isolated case
  - But a million medical records could lead to a cure

The Failure problem

- 5 commodity machines cheaper than one large machine
- If availability of large machine is 99.9% it will fail one day per year
- 5 machines will fail 5 days per year.
- Systems need to incorporate this downtime
- Imagine no Google for 5 days a year.

The problem of Scale

- “Anything that can go wrong will go wrong” - Murphy
- At 10000+ machines, If one fails everyday, A lot of failures will happen time and again
- At large scale. Small error rates manifest themselves
- **Amplification problem**

The problems to be solved

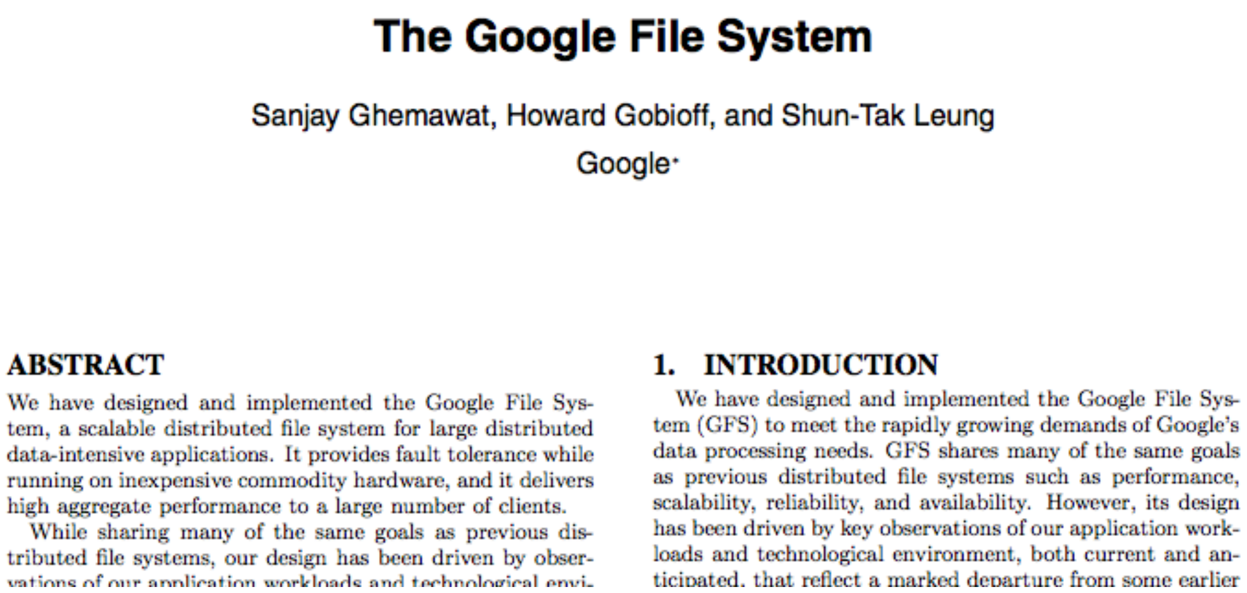
- How to store all that data at reasonable cost
- Process all the stored data

The solutions

- How to store all that data at reasonable cost
  - **Hadoop Distributed File System i.e. HDFS**
- Process all the stored data
  - **Hadoop Map Reduce Framework i.e. MapR**

GFS and MapReduce

- Produced by Jeff Dean, Sanjay Ghemawat and others
- GFS - Google File System
- Yahoo! open sourced Hadoop
- Hadoop = HDFS(GFS) + MapReduce
- Comes under the Apache License
- Hadoop was the name of Doug Cutting(Hadoop Inventor)'s kids doll; which was an elephant



The Google File System

Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung

Google*

ABSTRACT

We have designed and implemented the Google File System, a scalable distributed file system for large distributed data-intensive applications. It provides fault tolerance while running on inexpensive commodity hardware, and it delivers high aggregate performance to a large number of clients.

While sharing many of the same goals as previous distributed file systems, our design has been driven by observations of our application workloads and technological envi-

1. INTRODUCTION

We have designed and implemented the Google File System (GFS) to meet the rapidly growing demands of Google's data processing needs. GFS shares many of the same goals as previous distributed file systems such as performance, scalability, reliability, and availability. However, its design has been driven by key observations of our application workloads and technological environment, both current and anticipated, that reflect a marked departure from some earlier

Proceedings of the 19th ACM symposium on Operating systems principles
Over 5000 citations

What is ‘Distributed’?

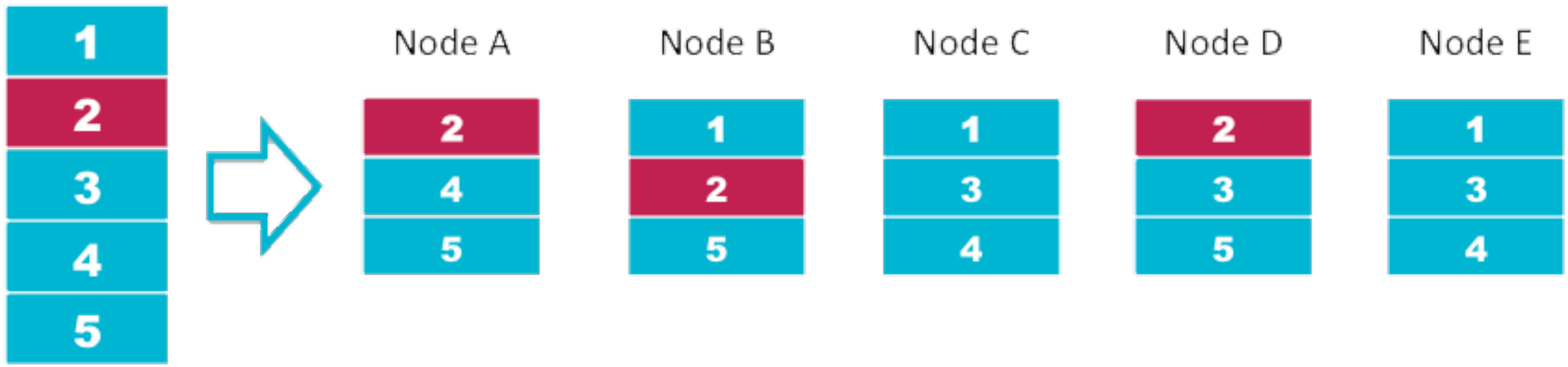
- Nothing is more important than anything else
  - No master, no slave, everyone is equal
- A colony of ants
- Parallel and distributed are different things (in some ways)
- Like workers lifting rocks. Expendable!
- In pioneer days they used oxen for heavy pulling, and when one ox couldn’t budge a log, We didn’t try to grow a larger ox

Key ideas

- Files are stored as chunks of 64 MB each (configureable)
- Reliability through replication.
  - Each chunk is replicated across 2+ chunkservers
- Single master to co ordinate access, keep metadata
  - Simple centralised management.
- No data caching
  - Little benefit due to large datasets, streaming reads.

In a nutshell

HDFS Data Distribution



Input File

Input File	Node A	Node B	Node C	Node D	Node E
1	2	1	1	2	1
2	4	2	3	3	3
3	5	5	4	5	4
4					
5					

More

- Replication is rack aware
- Hence, Fault tolerance is extremely high
- Supports scale of 10,000+ machines
- Allows for building higher degrees of abstraction
- Yahoo! - 4000 Nodes, Facebook - 2300 Nodes

Map Reduce

- Hopes for a fault tolerant file system underneath
- Applies computation to the data at the node
- No need to bring data to one place
- Send the computation to where the data resides
- Exploits the associative property of Mathematics

Map Reduce

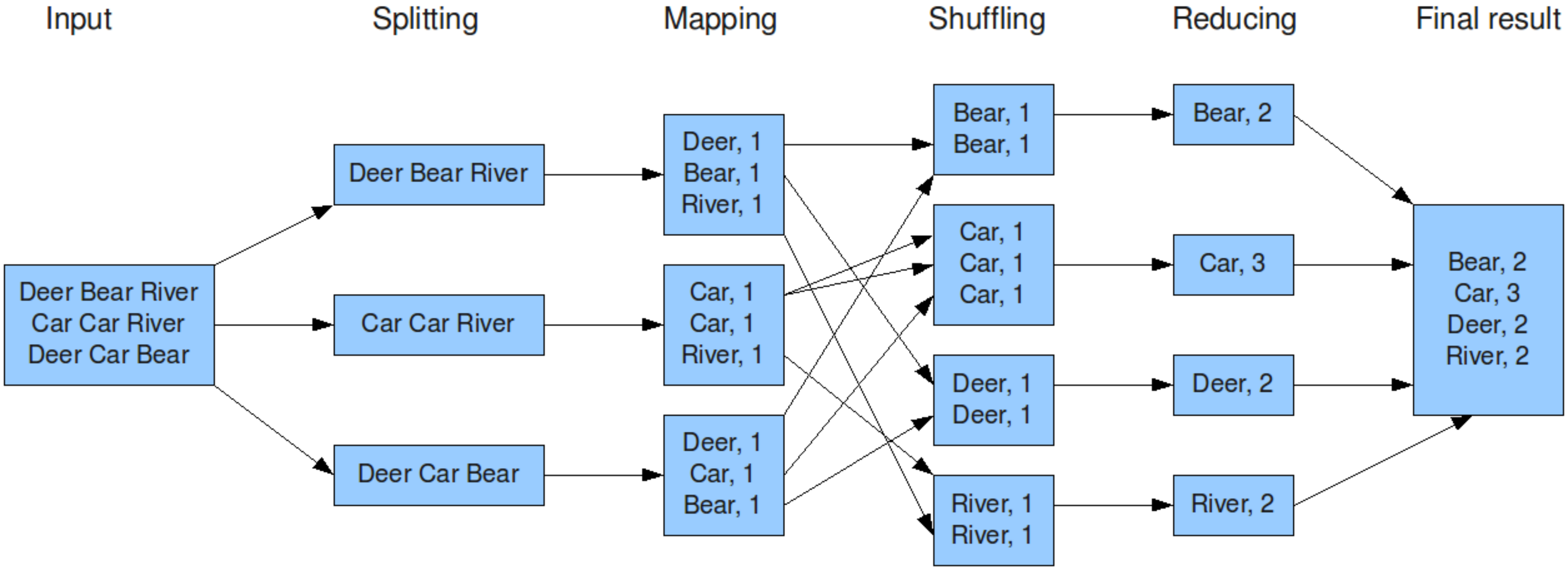
- Map is a one-to-one operation
- For some ‘x’, Apply a function, f(x) = y

- Reduce collects all y’s to give a result
- count(y) = n

- All maps are done in parallel, reduce is done in parallel over similar data items.

Illustration

The overall MapReduce word count process



Input

Deer Bear River
Car Car River
Deer Car Bear

Splitting

Deer Bear River

Car Car River

Deer Car Bear

Mapping

Deer, 1
Bear, 1
River, 1

Car, 1
Car, 1
River, 1

Deer, 1
Car, 1
Bear, 1

Shuffling

Bear, 1
Bear, 1

Car, 1
Car, 1
Car, 1

Deer, 1
Deer, 1

River, 1
River, 1

Reducing

Bear, 2

Car, 3

Deer, 2

River, 2

Final result

Bear, 2
Car, 3
Deer, 2
River, 2

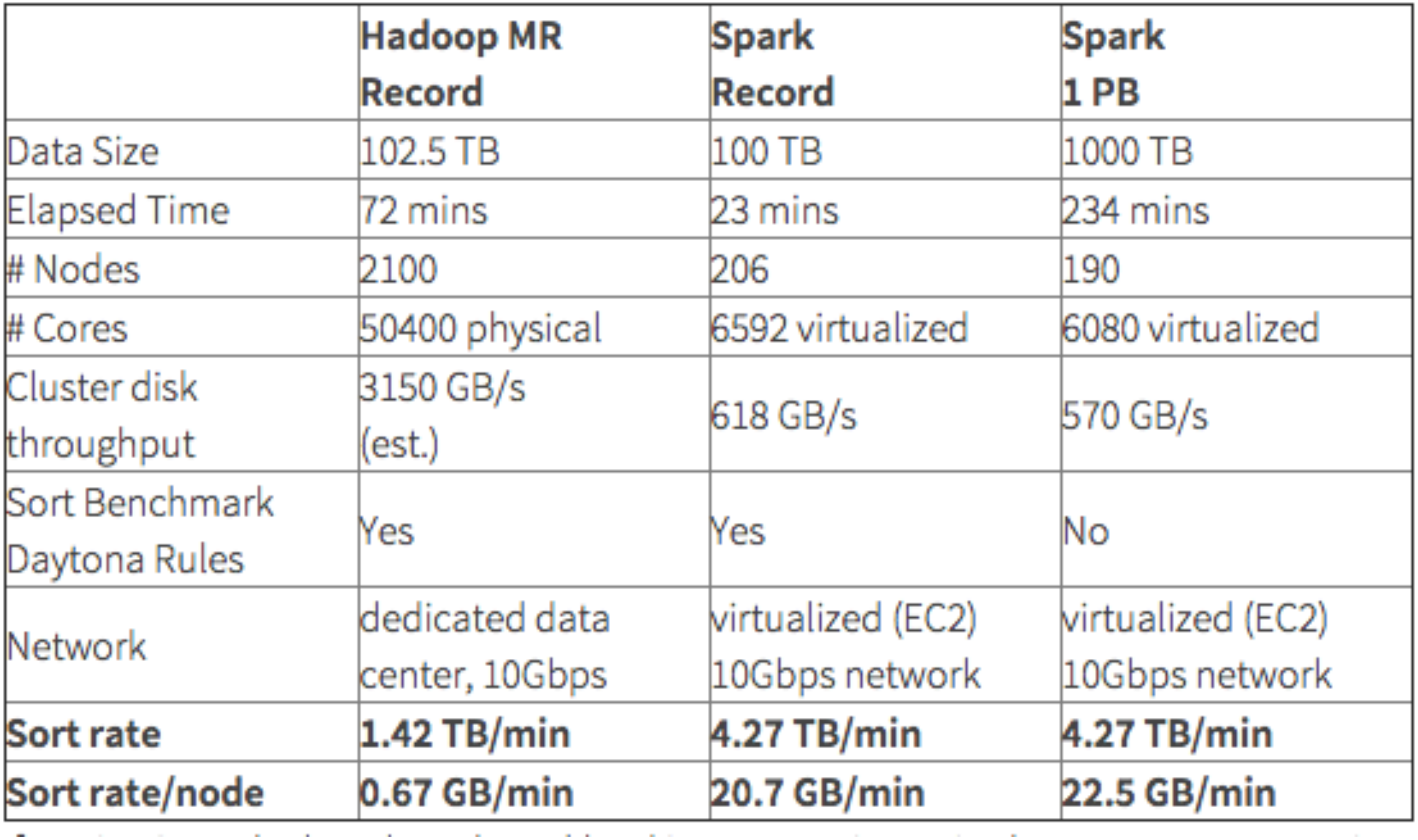
What happens when things break?

- Failed computations are re-run
- If one computer is slow, Its operations are given to others
- If a hard disk fails. Replace disk, data will be copied back

Next Generation

- 100’s of frameworks built on top of HDFS
- Cloudera offers a one-click solution for $$$$. Hortonworks is open-sourced
- Spark is the most upcoming project in the Apache group of projects
- Remember MapReduce had sorted 102 TB in 72 minutes using 2100 machines?

Where things are going



	Hadoop MR Record	**Spark Record**	**Spark 1 PB**
Data Size	102.5 TB	100 TB	1000 TB
Elapsed Time	72 mins	23 mins	234 mins
# Nodes	2100	206	190
# Cores	50400 physical	6592 virtualized	6080 virtualized
Cluster disk throughput	3150 GB/s (est.)	618 GB/s	570 GB/s
Sort Benchmark Daytona Rules	Yes	Yes	No
Network	dedicated data center, 10Gbps	virtualized (EC2) 10Gbps network	virtualized (EC2) 10Gbps network
Sort rate	**1.42 TB/min**	**4.27 TB/min**	**4.27 TB/min**
Sort rate/node	**0.67 GB/min**	**20.7 GB/min**	**22.5 GB/min**